REMARKS

Claims 1-28 were pending in the application. Claims 1, 3, 4, 10, 11, 13, 14, 16, 17, 23, 24, 26, 27, and 28 were rejected under 35 U.S.C. § 102(a) as anticipated by U.S. Patent 6,108,715 to Leach, et al. ("Leach"). Claims 2 and 15 were rejected under 35 U.S.C. § 103(a) over Leach in view of U.S. Patent 5,590,327 to Biliris, et al. ("Biliris"). Claims 5 and 18 were rejected under 35 U.S.C. § 103(a) over Leach in view of U.S. Patent 5,504,901 to Peterson ("Peterson"). Claims 6, 7, 19, and 20 were rejected under 35 U.S.C. § 103(a) over Leach in view of Peterson and further in view of U.S. Patent 6,381,735 to Hunt ("Hunt"). Claims 8 and 21 were rejected under 35 U.S.C. § 103(a) over Leach in view of U.S. Patent 6,263,379 to Atkinson, et al. ("Atkinson") and further in view of U.S. Patent 5,592,588 to Reekes, et al. ("Reekes"). Claims 9 and 22 were rejected under 35 U.S.C. § 103(a) over Leach in view of U.S. Patent 6,336,148 to Doong, et al. ("Doong"). Claims 12 and 25 were rejected under 35 U.S.C. § 103(a) over Leach in view of U.S. Patent 5,838,971 to Stadler, et al. ("Stadler"). The applicant requests reconsideration of the rejections in light of the following remarks.

1. Response to Rejections under Section 102

Claims 1, 3, 4, 10, 11, 13, 14, 16, 17, 23, 24, 26, 27, and 28 were rejected as anticipated by Leach. The applicant respectfully traverses the rejection.

Claim 1 specifically requires that a marshalling function take as input data and a *type object*, where the type object describes a data type. The claims recite that the type object is dynamically generated by calling a type creation function at runtime.

The Examiner finds in Leach a marshalling function that takes as input a type object that is dynamically generated by calling a type creation function at runtime at column 2, lines 45-57 and in figure 2. What column 2, lines 45-57 and figure 2 actually disclose is invoking a remote procedure using conventional marshalling. Leach describes the marshalling by stating that each method of a proxy object marshals its name and its input parameters into an interprocess communications message (2:65-67). Leach does not disclose or suggest a marshalling function that takes as input (i) data and (ii) a type object, nor does Leach disclose calling a type creation function at runtime to create the type object. Thus, the cited portions of Leach do not teach a marshalling function that takes as input a type object that is dynamically generated by calling a type creation function at runtime, and the rejection based on Leach should be withdrawn.

Claim 4 requires that the type object input to the marshalling function be a parameterized type object. Leach describes a signature table that contains call and return signatures for a real object (7:54-62). Code for a call signature can copy data and parameters that the real method of the real object needs to execute (7:62-8:2). Leach does not disclose or suggest using a parameterized type object as an input to a marshalling function, and claim 4 therefore is allowable over Leach. Claim 17 includes limitations corresponding to those of claim 4 and is allowable for at least the same reasons.

Claim 10 recites the method of claim 1 in which the type creation function is also called in the second process to create a second instance of the type object. Because Leach does not disclose or suggest calling a type creation function or creating a type object, claim 10 is allowable over Leach. Claim 23 includes limitations corresponding to those of claim 10 and is allowable for at least the same reasons.

Claims 3 and 11 depend directly from claim 1, and are allowable for at least that reason.

Independent claims 13, 14, and 26 include limitations corresponding to those of claim 1 and are allowable for at least that reason. Claims 16 and 24 depend directly from claim 14 and are allowable for at least that reason.

2. Response to Rejections under Section 103

Claims 2 and 15 were rejected as unpatentable over Leach in view of Biliris. The applicant respectfully traverses the rejection. Claim 2 depends directly from claim 1, and claim 15 depends directly from claim 14. The examiner does not appear to contend that Biliris suggests or discloses a marshalling function that takes as inputs data and a type object. Claims 2 and 15 are therefore allowable for at least the same reasons as claim 1.

Claims 5 and 18 were rejected as unpatentable over Leach in view of Peterson. The applicant respectfully traverses the rejection. The Examiner finds an offset element identifying a location in memory relative to data that is communicated to a second process in Peterson at column 4, lines 35-49. What Peterson actually teaches is an offset pointer identifying a location in memory relative to an entry point address of a procedure or relative to a memory location of an address of a called procedure. The cited portion of Peterson does not teach an offset element identifying a location in memory relative to data that is communicated to a second process, and the rejection based on Leach in view of Peterson should be withdrawn.

Claims 6, 7, 19, and 20 were rejected as unpatentable over Leach in view of Peterson and further in view of Hunt. The applicant respectfully traverses the rejection. The Examiner finds a parameterized type object describing a dynamically sized array and type parameters including data specifying a size of the dynamically sized array in Hunt at column 10, lines 17-39. What column 10, lines 17-39 discloses is an interface definition language (IDL) that enumerates the number and type of arguments passed through interface functions. An attribute can be attached to an interface or to a parameter of an interface function, and the attribute can specify the size of a dynamic array. The cited portion of Hunt does not teach a parameterized type object describing a dynamically sized array and type parameters including data specifying a size of the dynamically sized array, and the rejection of claims 6 and 19 based on Leach in view of Peterson and further in view of Hunt should be withdrawn.

The Examiner also finds a parameterized type object describing a dynamically typed pointer and type parameters including data identifying a second type object in Hunt at column 3, lines 59-62. What column 3, lines 59-62 actually discloses is using a stack pointer to identify a dynamically allocated object at the time the object is allocated. The cited portion of Hunt does not teach a parameterized type object describing a dynamically typed pointer and type parameters including data identifying a second type object, and the rejection of claims 7 and 20 based on Leach in view of Peterson and further in view of Hunt should be withdrawn.

Claims 8 and 21 were rejected as unpatentable over Leach in view of Atkinson and further in view of Reekes. The applicant respectfully traverses the rejection. The Examiner finds an encoded representation of a type-object description included in encoded data in Atkinson at column 76, lines 15-26 and at column 77, lines 45-60. What column 76, lines 15-26 and column 77, lines 45-60 actually disclose is marshalling an interface pointer into a stream (76:15-26). The marshalling function takes as inputs arguments that have associated types (77:45-60). The cited portion of Atkinson does not disclose an encoded representation of a type-object description included in encoded data.

The Examiner also finds an encoded representation of a type-object description included in encoded data in Reekes at column 13, lines 35-45. What column 13, lines 35-45 discloses is a structure describing a sound object. The cited portion of Atkinson does not disclose an encoded representation of a type-object description included in encoded data. Thus, the rejection based on Leach in view of Atkinson and further in view of Reekes should be withdrawn.

Claims 9 and 22 were rejected as unpatentable over Leach in view of Doong. The applicant respectfully traverses the rejection. The Examiner finds a type object including a limitation condition specifying a limitation on permissible values for data having a data type in Doong at column 5, lines 38-60. What column 5, lines 38-60 actually discloses is constraint-checking code that checks to see whether a variable sent from a client to a server has a legal value based on information to which the server has access (*e.g.*, based on the amount of money in a bank account). The cited portion of Doong does not disclose a type object including a limitation condition specifying a limitation on permissible values for data having a data type, and the rejection based on Leach in view of Doong should be withdrawn.

Claims 12 and 25 were rejected as unpatentable over Leach in view of Stadler. The applicant respectfully traverses the rejection. Claim 12 depends indirectly from claim 1, and claim 25 depends indirectly from claim 14. The examiner does not appear to contend that Stadler suggests or discloses a marshalling function that takes as inputs (i) data and (ii) a type object. Claims 12 and 25 are therefore allowable for at least the same reasons as claim 1.

3. Conclusion

The applicant submits that all claims are in condition for allowance and asks that all claims be allowed. Please apply any charges or credits to deposit account 06-1050.

Respectfully submitted,

Date: 14 July 04

Hans R. Troesch
Reg. No. 36,950

Fish & Richardson P.C.
500 Arguello Street, Suite 500
Redwood City, California 94063
Telephone: (650) 839-5070
Facsimile: (650) 839-5071

50223260.doc